Fall 2016 - CMU OCI

25m : 05s
to test end

☆ Arranging Coins

Ashley has *c* coins that she wants to arrange in an *m*-row staircase, meaning that every i^{th} row must have exactly *i* coins.

Complete the *arrangeCoins* function in your editor. It has *1* parameter: an array of *n* long integers, *coins*, where each element ($coins_i$) is a long integer representing some *c*. For each $coins_i$, your function must find m_i (the maximum number of full staircase rows that can be formed for the given value of *c*) and print it on a new line.

Input Format

The locked stub code in your editor reads the following input from stdin and passes it to your function:
The first line contains an integer, *n*, denoting the size of the *coins* array.
Each line *i* of the *n* subsequent lines (where $0 \le i < n$) contains a long integer describing element *i* in *coins*.

Constraints

- $1 \le n \le 10^5$
- $1 \le coins_i \le 10^{15}$, where $0 \le i < n$

Output Format

For each $coins_i$, your function must print an integer denoting the maximum value of m_i.

Sample Input 1

```
4
2
5
8
3
```

Sample Output 1

```
1
2
```

Fall 2016 - CMU OCI

25m : 05s
to test end

Explanation

coins = {2, 5, 8, 3}

1. *coins[0] = 2*
   The coins can form the following rows:

   ```
   ¤
   ¤
   ```

   Because the 2^{nd} row is incomplete, we print *1* on a new line.

2. *coins[1] = 5*
   The coins can form the following rows:

   ```
   ¤
   ¤ ¤
   ¤ ¤
   ```

   Because the 3^{rd} row is incomplete, we print *2* on a new line.

3. *coins[2] = 8*
   The coins can form the following rows:

   ```
   ¤
   ¤ ¤
   ¤ ¤ ¤
   ¤ ¤
   ```

   Because the 4^{th} row is incomplete, we print *3* on a new line.

4. *coins[3] = 3*
   The coins can form the following rows:

   ```
   ¤
   ¤ ¤
   ```

   Because the 2^{nd} row is complete, we print *2* on a new line.

Fall 2016 - CMU OCI

25m : 05s
to test end

We recommend you take a quick tour of our editor before you proceed. The timer will pause up to 90 seconds for the tour.

Start tour

Original code | Java 7

```
1  import ↔;
6
7  public class Solution {
8
9  /*
10  * Complete the function below.
11  */
12
13      static void arrangeCoins(long[] coins) {
14          if (coins == null || coins.length == 0) {
15              return;
16          }
17
18          for (int i = 0; i < coins.length; i++) {
19              long x = (long) Math.sqrt(coins[i] * 2);
20              if (coins[i] >= x * (x + 1) / 2) {
21                  System.out.println(x);
22              } else {
23                  System.out.println(x - 1);
24              }
25          }
26
27      }
28
29
30      public static void main(String[] args){↔}
45  }
```

Line: 9 Col: 1

Test against custom input

Run Code | Submit code & Continue

(You can submit any number of times)

Fall 2016 - CMU OCI

25m : 05s
to test end

About Privacy Policy Terms of Service

1

2

3

4

5